*ISSN 2782-2001*
*Системы анализа*
*и обработки данных*
*том 97, № 1, 2025, с. 7–26*

*http://journals.nstu.ru/vestnik*
*Analysis and data*
*processing systems*
*Vol. 97, No. 1, 2025, pp. 7–26*

ИНФОРМАЦИОННЫЕ
ТЕХНОЛОГИИ
И ТЕЛЕКОММУНИКАЦИИ

INFORMATION
TECHNOLOGIES
AND TELECOMMUNICATIONS

УДК 004.42

DOI: 10.17212/2782-2001-2025-1-7-26

# Разработка кросс-платформенного приложения для управления робототехническим комплексом с помощью Flutter*

**Д.С. КОЛТЫГИН[a], И.А. СЕДЕЛЬНИКОВ[b], О.К. КРУМИН[c]**

*РФ, 665709, Иркутская область, г. Братск, ул. Макаренко, 40, Братский государственный университет*

[a] *kds@brstu.ru* [b] *Ohtargil@yandex.ru* [c] *kruminok1977@gmail.com*

В статье приводится краткий анализ подходов к разработке программ управления технологическими процессами. Основные из них – это нативные приложения, а также кросс-платформенные и web-приложения. Выявлены их основные преимущества, недостатки, особенности применения и разработки. Изложено описание кросс-платформенной технологии Flutter, примененной при разработке систем управления робототехническим комплексом, включающее ее архитектуру, особенности языка программирования Dart и сложности, возникающие при реализации функций программы и связанные с обеспечением работоспособности программного обеспечения на различных операционных системах. Фреймворк Flutter поддерживает программирование для операционных систем семейств Windows, Linux, Mac, IOS и Android, а также для web-приложений. Представлена методика разработки приложения на примере робототехнического комплекса с несколькими роботами, подключенными к контроллеру на базе платы Arduino Mega, и различные средства управления. Описана структура робототехнического комплекса и разработанной программы, приведены внешний вид программы на разнообразных платформах и краткое описание ее работы. Отдельно рассмотрен вопрос управления правами пользователей с аутентификацией непосредственно на контроллере, что позволяет выполнять аутентификацию пользователей на нескольких типах устройств. Приведены особенности реализации кросс-платформенных функций, связанные с одновременным использованием библиотек dart:io и dart:html в различных ситуациях на примере работы с последовательным портом. Сформированы алгоритмы данных процессов и обоснования их применения. На основании анализа данной и предыдущих работ приводятся преимущества и недостатки таких сред разработки, как Visual Basic, MIT App Inventor, ASP.Net, MAUI/Xamarin, Flutter, выявленные при создании комплекса программ.

**Ключевые слова**: кросс-платформенное приложение, веб-приложение, Flutter, Dart, Робот, система управления, алгоритм, разработка, SCADA, Arduino

* *Статья получена 10 ноября 2024 г.*

## ВВЕДЕНИЕ

В настоящее время в системах автоматического управления используется множество платформ: серверных, настольных, мобильных, а также большое количество операционных систем (ОС), таких как Windows, Linux, MacOS, Android, IOS с их версиями и вариациями. С недавних пор всё актуальнее встает вопрос перехода на отечественные платформы, например, Aurora, Astra Linux, RedOS и т. д. Некоторые организации в своей деятельности могут одновременно применять целый набор операционных систем, следовательно, программа управления должна работать на каждой из них. К такому выводу приходят многие крупные разработчики, например «МПС Софт» – создатель MasterSCADA, AdAstra – Trace mode и другие. При этом актуальным является выбор методов и средств проектирования системы управления.

Существует три основных подхода к разработке таких систем: создание нативных приложений для каждой ОС, создание web-приложения, создание кросс-платформенного приложения [1–4].

В табл. 1 приведен анализ основных параметров различных приложений.

*Таблица 1*

*Table 1*

**Анализ параметров приложений**

**Application parameter analysis**

| Вид приложения / Параметр | Нативные приложения | Web-приложение | Кросс-платформенное приложение |
|---|---|---|---|
| Установка | Требуется | Не требуется | Требуется |
| ОС | Для каждой ОС свое приложение | Любая | Любая из поддерживаемых средством разработки |
| Язык программирования | Для каждой ОС свой язык | Единый язык для всех ОС | Единый язык для всех ОС, поддерживаемых средством разработки |
| Скорость разработки и отладки | Высокая – для отдельного приложения, низкая – в совокупности | Высокая | Средняя. Требуется оптимизация некоторых задач под разные ОС |
| Обновление на стороне пользователя | Повторная установка или обновление | Не требуется | Повторная установка или обновление |
| Скорость работы | Высокая | Низкая | Средняя |
| Отличия в работе | Могут быть отличия в интерфейсе и функционале на различных устройствах | Одинаковый функционал и интерфейс на любом устройстве | Одинаковый функционал и интерфейс могут иметь отличия, связанные с особенностями ОС |

Отсюда можно сделать вывод, что предпочтительнее использовать кросс-платформенное или web-приложение. Они обладают большей универсальностью по сравнению с нативными, поэтому требуют меньше времени на разработку и сокращают затраты на поддержание, обновление и модернизацию.

В статье рассмотрена методика применения фреймворка Flutter, поскольку он позволяет собрать одновременно кросс-платформенное приложение и web-приложение. В то же время он поддерживает наибольшее количество операционных систем из аналогичных фреймворков и, следовательно, покрывает самый широкий круг задач.

Стоит отметить, что в русскоязычной научной среде вопрос кросс-платформенной разработки раскрыт слабо. В основном описаны либо только мобильные, либо web-решения, а сфера систем управления практически не освещается.

Анализируя статьи [5–7], можно обратить внимание на то, что большинство авторов ориентируются именно на мобильные платформы и мало уделяют внимания настольным системам. Однако в нашей стране много предприятий, переходящих на отечественные ОС на базе Linux или имеющих такие системы на Windows, требующие модернизации.

## 1. ПОСТАНОВКА ЗАДАЧИ

Цель работы – разработать систему управления движением робототехнического комплекса (РТК) на базе двух роботов-манипуляторов, в котором в качестве основного средства управления следует использовать кросс-платформенное приложение. Такой тип приложений в настоящее время редко применяется в системах управления, из-за чего имеется немного информации о трудностях, возникающих в процессе разработки, и путях их решения. Одной из таких проблем является необходимость обращения к нескольким библиотекам при реализации одной функции вследствие отсутствия возможности использования единой библиотеки на всех требуемых платформах. Особенно остро этот вопрос стоит при реализации web-приложения из-за обращения внутри него к разным базовым сборкам dart:io и dart:html, что вызывает конфликт и ведет к невозможности запуска приложения, а в некоторых случаях даже к компиляции. Например, такая ситуация возникает при попытке подключения к последовательному порту для управления контроллером. В ряде случаев возможно выбрать библиотеку в зависимости от платформы, например, при вызове функции. Однако если необходимо создать объект до использования функции, то такой метод не будет реализован, и, следовательно, необходимо реализовать другие методы решения.

Кроме того, в системе управления есть сторонние устройства и возможен многопользовательский режим работы, поэтому следует разработать метод разделения прав пользователей и управления привилегиями.

Авторы предлагают оригинальное решение и пути для разработчиков современного программного обеспечения кросс-платформенных технологий на примере системы управления РТК.

## 2. ОПИСАНИЕ ФРЕЙМВОРКА FLUTTER

Flutter – это открытая кросс-платформенная технология от Google, предназначенная для создания приложений для iOS, Android, Web и настольных операционных систем из единой кодовой базы. Она использует язык программи-рования Dart и представляет свой собственный набор виджетов для создания интерфейса. Сейчас Flutter поддерживает iOS, Android, Web, Windows, Linux, macOS, а с недавнего времени WebAssembly и «встраиваемые устройства (embedded devices)» [8].

Архитектура Flutter состоит из трех слоев: Framework, Engine и Embedder – их структура и компоненты приведены на рис. 1.

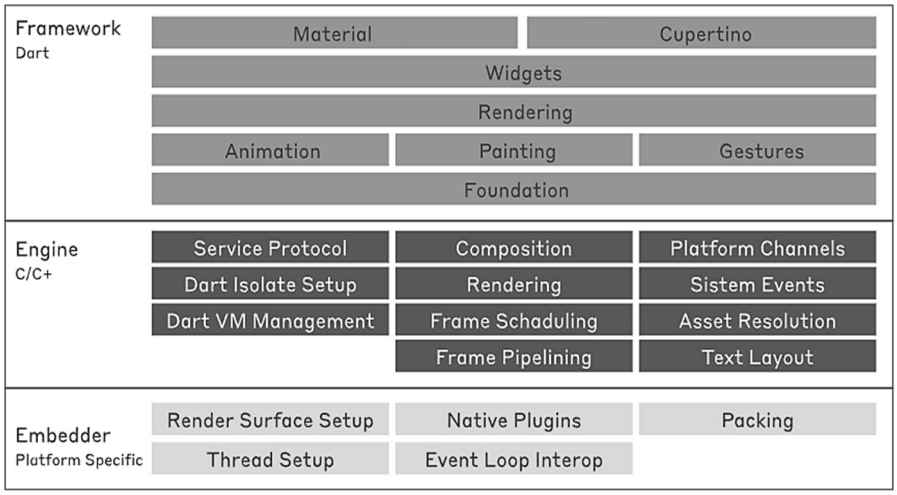



*Рис. 1*. Архитектура Flutter

*Fig. 1*. Flutter architecture

В общих чертах слой Framework, состоящий из кода на языке Dart, отвечает за пользовательский интерфейс и его взаимодействие с приложением, Engine – за логику рендера и обеспечивает низкоуровневую реализацию основных API Flutter, а Embedder интегрирует движок Flutter в различные платформы [9].

В процессе рендера Flutter имеет собственный движок рендеринга Skia Graphics Engine. С его помощью фреймворк рисует каждый элемент UI (виджет) на холсте, предоставляемый любой платформе.

При компиляции Flutter использует язык Dart, который компилируется в нативный код, что позволяет взаимодействовать с платформой без моста JavaScript (как в некоторых других кросс-платформенных решениях) и обеспечивает производительность, близкую к нативной [8].

### 2.1. ЯЗЫК DART И ЕГО ОСОБЕННОСТИ

Dart – современный объектно ориентированный язык программирования, разработанный Google. Он может использоваться для разработки как на стороне клиента, так и на стороне сервера. Это упрощает процесс обучения и разработки, поскольку разработчики могут использовать один язык для всего

стека приложений. Dart поддерживает асинхронное программирование, что очень важно для современной разработки приложений. Это позволяет приложениям на Flutter выполнять длительные задачи, такие как получение данных из сети, не блокируя UI и не влияя на отзывчивость приложения. Еще одно из достоинств Flutter-сообщества – это экосистема библиотек, т. е. наличие единого сервиса, на котором публикуются сторонние библиотеки. Любой желающий может создать свой пакет и разместить его на pub.dev.

### 2.2. ОГРАНИЧЕНИЯ ИСПОЛЬЗОВАНИЯ КРОСС-ПЛАТФОРМЫ И ПУТИ ИХ РЕШЕНИЯ

Стоит отметить и недостатки кросс-платформенной разработки. Часть из них преодолима, другую же часть остается пока только принять как данность. Приложения Flutter могут иметь больший размер файла по сравнению с нативными приложениями [10]. Не все функции поддерживаются операционной системой либо требуют применения различных библиотек. Эта особенность будет рассмотрена в статье отдельно. Однако стоит отметить, что Flutter – относительно новый фреймворк. Его прототип появился в 2015 году, но первый стабильный релиз Flutter 1.0 появился только в декабре 2018 года. С тех пор регулярно выходят новые версии, расширяющие функциональность и исправляющие имеющиеся недостатки. На момент публикации настоящей статьи последней стабильной версией является Flutter 3 с подверсией 3.24.3, вышедшей 12 сентября 2024 года.

## 3. СТРУКТУРА СИСТЕМЫ УПРАВЛЕНИЯ

При разработке приложения был использован РТК, включающий в себя два манипулятора МП-11 и транспортер. Управление оборудованием осуществляет контроллер, специально разработанный для этого РТК на основе платы Arduino Mega с bluetooth-модулем hc-05 и беспроводным NRF24L01+PA+LNA. Манипуляторы подключаются парой шлейфов, один из которых служит для подачи управляющих сигналов, а второй – для сбора данных с датчиков. Управляющие команды возможно подавать с помощью пульта управления по радио или с инфракрасного пульта управления. Основным средством взаимодействия оператора и РТК предполагается программа, предоставляющая доступ с любого типа устройств, таких как мобильные телефоны и планшеты, подключаемые через bluetooth, компьютеры – через USB или же через web-браузер. Для отображения состояния манипуляторов в программу может передаваться видео с камеры. Структурная схема приведена на рис. 2.

В контроллер записана специально разработанная программа низкого уровня, которая осуществляет управление функциями принятия, обработку и отправку команд, управление подключенными компонентами (реле, платы расширения и т. д.). Более подробно о создании контроллера и его программной составляющей написано в статье [11].

Основные возможности программы включают в себя:

– автоматическое и ручное управление РТК;
– локальное и удаленное управление РТК;

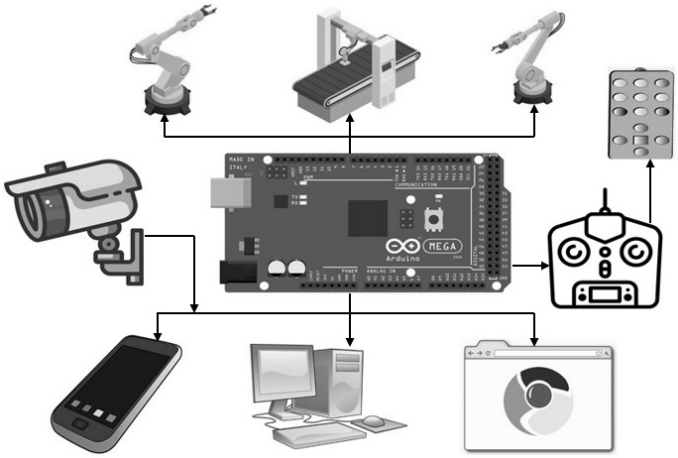


*Рис. 2.* Структурная схема робототехнического комплекса

*Fig. 2*. Block diagram of the robotics complex

– режим имитации и управления непосредственно РТК;
– управление с помощью кнопок, текстовых и голосовых команд;
– создание управляющих программ в режиме обучения.

### 3.1. СТРУКТУРА РАЗРАБАТЫВАЕМОГО ПРИЛОЖЕНИЯ

Для реализации представленных возможностей разработана следующая структура приложения (рис. 3).

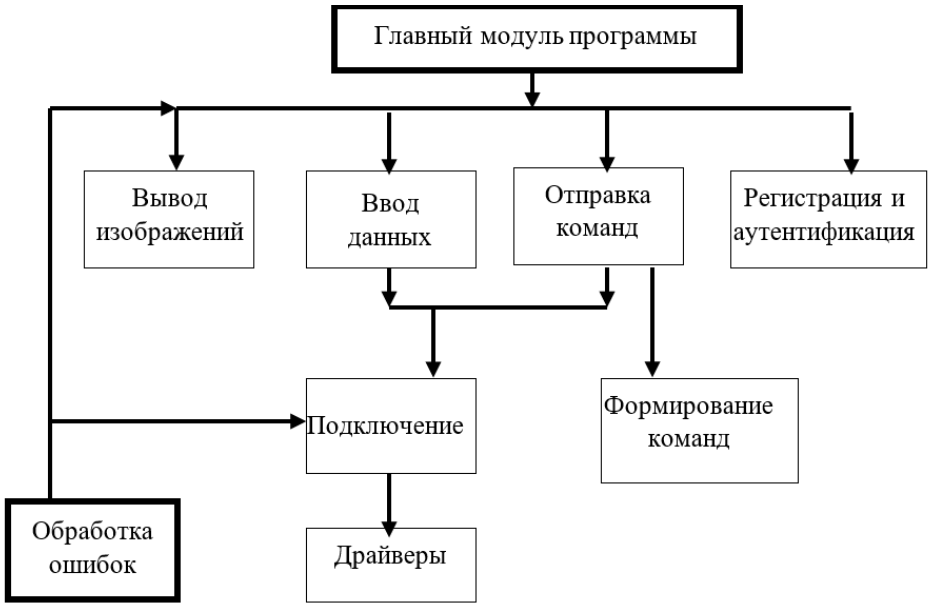



*Рис. 3.* Структурная схема приложения

*Fig. 3*. Block diagram of the application

Структура приложения представлена четырьмя слоями.

• Первый – главный модуль программы, осуществляет управление всеми функциями.

• Второй слой выполняет функции, отвечающие за решение основных задач программы, таких как ввод и вывод данных, интерфейс оператора, его доступ к программе и управление правами.

• Третий слой – вспомогательные функции, например сопряжение устройств и преобразование команд.

• Четвертый слой формально не относится к самой программе, но без него невозможно выполнить сопряжение, – это драйверы.

Обособленно стоит обработка ошибок. Она выполняется во всех функциях, где это необходимо.

*Модуль вывода изображения*. Отображение состояния робота-манипулятора осуществляется с помощью IP-камер или статическими изображениями положения. Поскольку такие роботы имеют некоторое количество состояний, для каждого из них заранее подготовлены картинки и загружены в ресурсы программы для быстрого доступа к ним.

*Модуль ввода данных*. Поскольку используется большое количество входных данных, то и применяется несколько вариантов их получения: ввод текстовой информации в виде команд в специальные поля, получение информации из состояния кнопок, заполнение многострочного текстового поля содержимым файла.

*Модуль отправки команд*. Вывод осуществляется отправкой сформированной команды на COM-порт. Дальнейшая их интерпретация и обработка происходит в контроллере. Помимо этого, список команд выводится в специальное поле и может быть сохранен в файл в любом режиме управления.

*Модуль регистрации и аутентификации пользователей.* Для управления правами реализована функция аутентификации пользователя и его регистрация в двух формах – локально и на сервере.

*Модуль подключения*. Программа обеспечивает подключение управляющего устройства к контроллеру по одному из трех каналов передачи: USB, bluetooth или сеть. Для это используется требуемый пакет драйверов. В данном случае драйверы устанавливаются только при USB-подключении на компьютер или web-сервер.

*Модуль формирования команд*. Поскольку в программе используется множество вариантов подачи команд (с использованием кнопок, текста и голоса) в разных форматах согласно языку команд [12], разработана функция, приводящая их к виду, удобному для передачи и приема контроллером.

*Модуль обработки ошибок*. Обработка ошибок производится на различных этапах выполнения программы. В каждом случае прописаны действия при прерывании либо невыполнении текущего процесса. Отдельно на основании опроса датчиков и состояния передачи производится анализ работы манипулятора и контроллера.

## 4. ОПИСАНИЕ СИСТЕМЫ УПРАВЛЕНИЯ ПРАВАМИ ПОЛЬЗОВАТЕЛЕЙ

Аутентификация пользователей – обязательная функция любой системы управления. В настоящее время для управления РТК применяется целый набор средств управления: инфракрасный или радиопульт, программы для компьютера и Android, web-приложение. Следовательно, необходимо обеспечить проверку прав пользователей, при этом оператор вводит единый логин и пароль на любой из них. Также важно исключить возможность нескольких подключений к РТК одновременно во избежание совместных попыток управления, способных привести к нештатным ситуациям. Для обеспечения корректной работы достаточно двух типов прав пользователя – с возможностью управления роботами либо только взаимодействием с моделью (режим имитации). Исходя из этого решено создать по группе пользователей на каждый тип, а аутентификацию проводить непосредственно на контроллере как конечном устройстве управления для всех систем.

На рис. 4 приведен алгоритм процесса аутентификации.

На средство управления при подключении передается один из трех сигналов: отказ в подключении (нет пользователя с таким логином/паролем); устройство занято (уже подключен другой пользователь); разрешение на подключение.

Кроме того, существует логин и пароль администратора, который служит для принудительного подключения к контроллеру с отключением существующего пользователя. Этот же пароль указан в локальных средствах управления (пультах). Из достоинств такого подхода можно выделить отсутствие необходимости создавать пользователя в каждом средстве управления и сокращение базы пользователей, поскольку в ней будет храниться ограниченный список тех, кто допущен непосредственно к управлению РТК.

## 5. ОПИСАНИЕ ПРИЛОЖЕНИЯ

Разработанное приложение поддерживает три основных режима работы: ручное управление, автоматическое управление, голосовое управление. Для каждого режима сформирован свой экран, выбор которого выполняется кнопками в верхней части формы. Приложение позволяет управлять как непосредственно РТК, так и его моделью в режиме имитации, установленном в качестве режима по умолчанию. Для подключения к контроллеру необходимо нажать соответствующую кнопку и ввести логин и пароль пользователя. Во всех режимах поддерживается функция записи выполняемых команд, используемая для создания управляющих программ. В результате записи формируется текстовый файл и сохраняется в выбранную директорию.

В ручном режиме управление (рис. 5) движением роботов осуществляется с помощью кнопок, расположенных на экране. Положение роботов можно отслеживать как на изображениях, так и по текстовым командам, отображаемым в соответствующих окнах. Режим автоматического управления (рис. 6) служит для формирования и выполнения управляющих программ. Для этого на экране располагается окно, в котором отображаются команды, кнопки управления, загрузки из файла и сохранения в файл. Режим голосового управления в целом

соответствует ручному, только вместо нажатия кнопок необходимо произнести в микрофон соответствующую команду.


*Рис. 4.* Алгоритм процесса аутентификации

*Fig. 4*. The algorithm of the authentication process

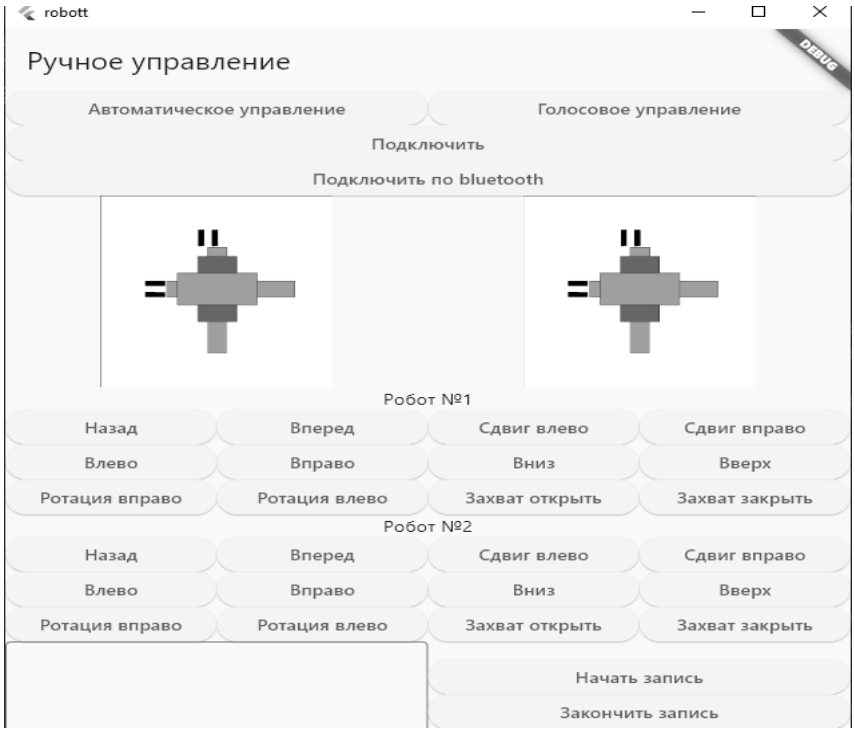



*Рис. 5.* Экран «Ручное управление» на ПК

*Fig. 5*. The "Manual control" screen on a PC

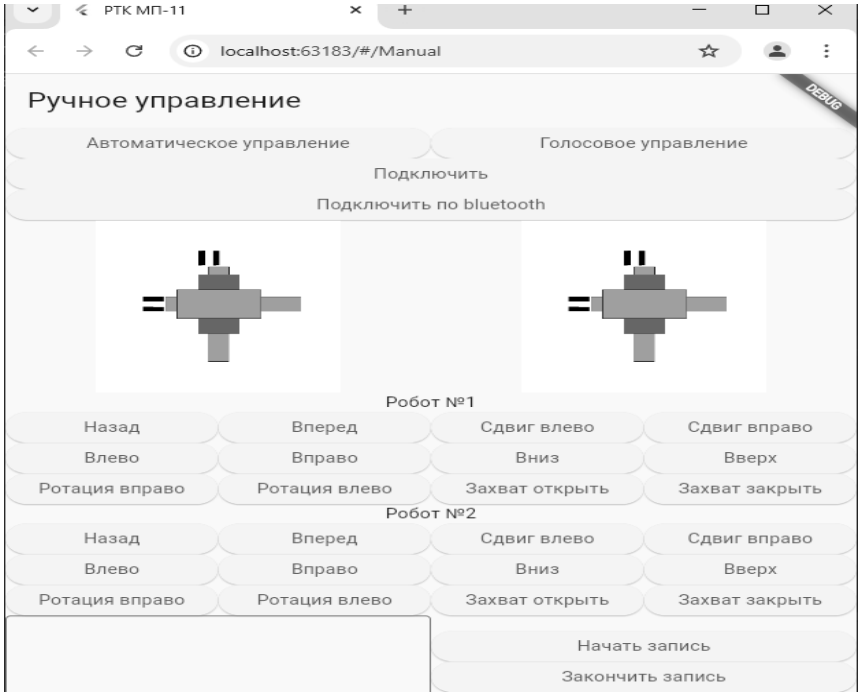



*Рис. 6.* Экран «Автоматическое управление» в web-браузере

*Fig. 6*. The "Automatic control" screen in a web browser

## 6. ОСОБЕННОСТИ РЕАЛИЗАЦИИ ФУНКЦИЙ ПРИ ВЗАИМОДЕЙСТВИИ С ПЛАТФОРМОЙ

Одна из основных причин выбора Flutter – возможность создания приложений, работающих на нескольких платформах с единой кодовой базой. В большинстве случаев весь требуемый функционал реализуется с использованием базовых элементов. Однако сложности возникают, когда необходимо обеспечить взаимодействие с платформой, например работу с портами, как в рассматриваемой системе управления. Здесь можно выделить несколько основных проблем:

1) на одном устройстве есть варианты использования функций, которых нет на другом, что связано с конкретным оборудованием (например, с наличием или отсутствием камеры);

2) другая проблема заключается в самой платформе. Flutter Mobile и Desktop установлены непосредственно в устройстве, что означает наличие возможности выполнения операций ввода / вывода, чтение, запись и создание локальных файлов. Flutter Web привязан к HTML-документу и выполняет определенные операции, такие как переход на другие веб-сайты, установка файлов cookie и изменение DOM.

Первая проблема решается проверкой оборудования и запретом на использование функций или обработкой ошибок выполнения. Вторая же требует более сложного решения. Причина в том, что приложение опирается на разные базовые библиотеки: Flutter для мобильных устройств и компьютеров использует dart:io, Flutter Web использует dart:html. Это приводит к тому, что если приложение одновременно использует dart:io и dart:html, то при выполнении будет выведено сообщение об ошибке либо возникнет ошибка при компиляции. Можно выделить три основных метода решения: 1) сборка отдельных проектов для мобильной / настольной платформы и для web; 2) условный импорт пакетов; 3) создание промежуточного модуля (заглушки). Основной недостаток первого метода заключается в том, что отчасти теряется смысл кросс-платформенной разработки, поскольку усложняется процесс исправления ошибок и поддержания кода. Недостаток второго метода заключается в том, что он работает только в случае, если пакет используется в изолированных функциях.

Ниже приведен фрагмент кода реализации условного импорта пакетов на примере функции сохранения набора команд в файл:

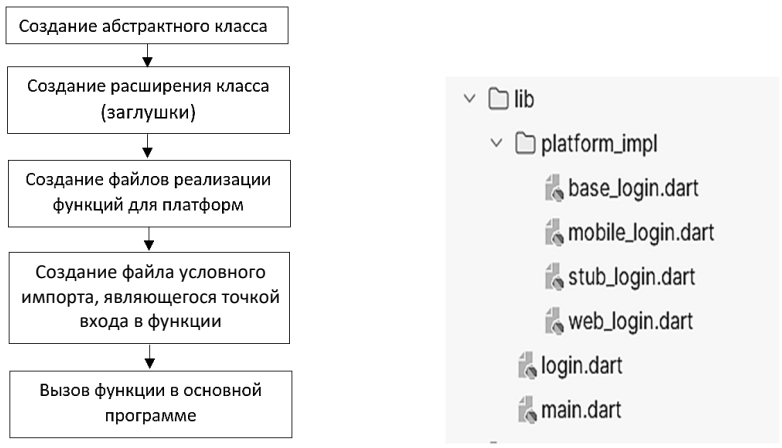
```
// запись в файл
Future<void> _saveTextToFile() async {
  //создание массива байт для сохранения в файл
  String myString = _nameController.text;
  Uint8List bytes = Uint8List.fromList(myString.codeUnits);
  //путь для сохранения
  String? outputFile = '';
  // WEB
  if (kIsWeb) {
    await FileSaver.instance.saveFile(
      name: 'New_program', // you can give the CSV file name here.
      bytes:  bytes,//bytes,
      ext: 'txt',
      mimeType: MimeType.text,
    );  }
//Android IOS
  if (Platform.isAndroid || Platform.isIOS) {
```

```
    if (!await FlutterFileDialog.isPickDirectorySupported()) {
      print("Picking directory not supported");
      return;    }
    final pickedDirectory = await FlutterFileDialog.pickDirectory();
    if (pickedDirectory != null) {
      final outputFile = await FlutterFileDialog.saveFileToDirectory(
        directory: pickedDirectory!,
        data: bytes,
        mimeType: "text/txt",
        fileName: "New_program",
        replace: true,
      );    }  }
  //windows linux
  else {
    outputFile = await FilePicker.platform.saveFile(
      dialogTitle: 'Please select an output file:',
      fileName: 'New_program.txt',
    );
    if (outputFile != null) {
      File file = File('${outputFile}');
      await file.writeAsString(_nameController.text);
    }  }}
```

В программе использован третий подход, описанный ниже [13].

Рассмотрим пример реализации промежуточного модуля, выполняющего функции подключения по USB, опроса состояния порта и отправки команд. На рис. 7, *а*, приведен алгоритм реализации третьего метода. Для реализации данных функций используют два пакета: serial.dart – для web и flutter_libserialport.dart – для остальных платформ.

При создании промежуточного модуля получим следующую структуру программных файлов (рис. 7, *б*).




*а* *б*

*Рис. 7.* Алгоритм метода «Создания промежуточного модуля» (*а*); структура программных файлов (модулей) (*б*)

*Fig. 7.* The algorithm of the "Creating an intermediate module" method (*a*); the structure of program files (modules) (*b*)

Первый файл, который необходимо создать, это base_login.dart, он содержит только код объявления абстрактного класса. Абстрактный класс – это класс, который

содержит абстрактные методы без их реализации, описывает общую структуру и определяет методы, которые должны быть реализованы любым подклассом [14].

Для реализации созданного класса необходимо сформировать расширение (подкласс) и связать его с основным классом. Реализация содержится в файле stub_login.dart, фрагмент кода приводится на рис. 8. В нем описаны все функции, которые могут быть выполнены при обращении к классу.

```dart
import 'package:robott/platform_impl/base_login.dart';
class LoginImpl extends BaseLogin {
 @override
  void login() {
    throw Exception("Stub implementation"); }
  void send(String str) {
    throw Exception("Stub implementation");}
  int stateport(){
    throw Exception("Stub implementation"); }
 }
```

*Рис. 8.* Фрагмент кода промежуточного модуля

*Fig. 8*. A code fragment of the intermediate module

Затем необходимо создать два файла: mobile_login.dart и web_login.dart. Фрагменты кода приведены на рис. 9 и 10. Названия функций в них должны быть одинаковы и соответствовать stub_login.dart. Эти файлы также содержат связь с абстрактным классом и пакеты для работы с портом на разных платформах – flutter_libserialport.dart в и serial.dart  соответственно.

```dart
import 'package:robott/platform_impl/base_login.dart';
import 'package:flutter_libserialport/flutter_libserialport.dart';
import 'package:flutter/foundation.dart';
class LoginImpl extends BaseLogin {
  final  SerialPort _loginLib;
  LoginImpl() : _loginLib =  SerialPort('COM3');//MobileLibForLogi
  SerialPort port= SerialPort('');// создание порта без  имени
  @override
  void login() async{
    String portname='';
    port.close();
    var availablePorts = SerialPort.availablePorts;
    for (final address in availablePorts) {
      portname = address;
    }
    port = SerialPort(portname);
    await port.openReadWrite();
    var config = port.config;
    config.baudRate = 9600;
    port.config = config;
    port.config.bits = 8;
    port.config.stopBits = 1;
    port.config.parity = 0;
    port.config = config;
    print(portname);
  }
  void send(String str) async{
       Uint8List uint8list = Uint8List.fromList(str.codeUnits);
    await port.write(uint8list);
  }
  //статус порта
  int stateport(){
```

*Рис. 9.* Фрагмент кода модуля функций для dart:io

*Fig. 9.* A fragment of the function module code for dart:io

```dart
import 'dart:html';
import 'package:serial/serial.dart';
import 'package:robott/platform_impl/base_login.dart';
import 'package:flutter/foundation.dart';
class LoginImpl extends BaseLogin {
   SerialPort? _port;
   @override
  Future<void> _openPort() async {//  подключение обработка
     await _port?.close();
    final port1 = await window.navigator.serial.requestPort();
    await port1.open(baudRate: 9600);
    _port = port1;
  }
  Future<void> _writeToPort(Uint8List data) async {// отправка команды обработка
    if (data.isEmpty) {
      return;
    }
    final port = _port;
    if (port == null) {
      return;
    }
    final writer = port.writable.writer;
    await writer.ready;
    await writer.write(data);
    print(data.toString());
    await writer.ready;
    await writer.close();
  }
    void login() async{// подключение основная
     await _openPort();
  }
```

*Рис. 10.* Фрагмент кода модуля функций для dart:html

*Fig. 10*. A fragment of the function module code for dart:html

Теперь необходимо создать файл, который будет осуществлять условный импорт созданных реализаций функций и предоставит единственную точку входа login.dart. Код файла со ссылкой на stub_login.dart отображен на рис. 11.

```dart
import 'platform_impl/stub_login.dart'
if (dart.library.io) 'platform_impl/mobile_login.dart'
if (dart.library.html) 'platform_impl/web_login.dart';
class Login {
  final LoginImpl _login;
  Login() : _login = LoginImpl();
  void login() {
    _login.login();
  }
  void send(String str) {
    _login.send( str);
  }
    int stateport(){
    return  _login.stateport( );
  }
}
```

*Рис. 11.* Фрагмент кода модуля login.dart

*Fig. 11*. A code fragment of the login.dart module

В основном файле программы main.dart производится импорт пакета login.dart, предоставляющий точку входа. Вызов функций осуществляется через этот пакет. Пример обращения к функции показан ниже:

```
    import 'package:robott/login.dart';
//подключение к контролеру
void _conect() async { login.login(); }
// отправка команды
void _send(String str) async {  login.send(str);}
```

## 7. СРАВНЕНИЕ ПРОГРАММ, СОЗДАННЫХ В ХОДЕ ИССЛЕДОВАНИЯ

В рамках исследования для данного РТК был разработан ряд программ [11, 15, 16]: для Windows – с использованием языка Visual Basic, для Android – с помощью MIT App Inventor, для web-приложения – ASP.Net, для кросс-платформенного приложения – MAUI/Xamarin и для приложения, описанного в настоящей статье, – фреймворк Flutter. Все эти программные средства представляют разные подходы к созданию системы управления, обладающие разным функционалом и возможностями. Анализируя процесс создания программ, авторы статьи сделали следующие выводы (табл. 2).

*Таблица 2*

*Table 2*

**Анализ средств разработки**

**Analysis of the development tools**

| Среда | Недостатки | Преимущества |
|---|---|---|
| Visual Basic | Ориентировано на ОС Windows | Большое количество готовых решений для широкого спектра задач, справочной литературы, библиотек.<br>Удобный интерфейс и высокая скорость разработки |
| MIT App Inventor | Только для ОС Android.<br>Неудобство формирования и чтения кода программы.<br>Мало справочной информации и примеров.<br>Ограниченные функциональные возможности | Удобство проектирования интерфейса и отладки программы.<br>Удобство совместной работы над проектом, так как разработка ведется в web-среде по учетной записи |
| ASP.Net | Сложность реализации некоторых функций, так как обработка осуществляется на сервере, а не на локальном устройстве | Web-приложение является кросс-платформенным с доступом через браузер через сеть или через локальный сервер |

*Окончание табл. 2*

*End of the Tab. 2*

| Среда | Недостатки | Преимущества |
|---|---|---|
| MAUI/ Xamarin | Поддерживается небольшое количество ОС.<br>Сложности при реализации многих функций, связанные с тем, что фреймворк еще разрабатывается.<br>Мало справочной литературы и примеров | Кросс-платформенное решение |
| Flutter | Не все функции работают на различных ОС, либо требуется применение специальных алгоритмов (описанных в данной статье).<br>Интерфейс и обработка описываются в едином файле, что затрудняет чтение кода и работу с ним | Кросс-платформенное решение.<br>Удобный процесс создания интерфейса.<br>Большое количество доступных библиотек и решений с подробным описанием и примерами |

## ЗАКЛЮЧЕНИЕ

В настоящее время Flutter редко применяется при разработке систем управления, особенно для настольных и серверных устройств, однако при разнообразии используемых платформ он мог бы стать отличным решением в сфере создания программного обеспечения. Одним из основных преимуществ данного фреймворка является очень высокая скорость работы.

Методики, описанные в статье, являются примером реализации создания программ управления РТК, однако они могут быть использованы и для других роботов и систем управления.

Возможность запуска Flutter-приложения на встраиваемых системах позволяет создавать пульты управления оборудованием с привычным интерфейсом.

Использование кросс-платформенного подхода дает возможность существенно сократить сроки разработки систем управления для разного типа аппаратуры, сделать программы универсальными, переносимыми на различные платформы и языки программирования.

Кросс-платформенное приложение при исследовании показало скорость работы, сопоставимую с windows-приложением и в среднем на 15 % больше, чем скорость отдельного web-приложения, сохранило свою функциональность на различных операционных системах (linux ubuntu-24.04.1, windows 10, android 14) и браузерах.

Применение промежуточного модуля для работы обеспечило надежную одновременную работу с библиотеками, обращающимися к различным

базовым сборкам dart:io и dart:html. Это решило основную проблему реализации функций подключения и обработки речи.

Метод аутентификации пользователя на контроллере позволил реализовать доступ по отпечатку и двухфакторную аутентификацию на высокопроизводительных устройствах управления и при этом сохранить локальный пульт с более высоким приоритетом.

## СПИСОК ЛИТЕРАТУРЫ

1. *Исаков Л.А.,* Инновационные подходы в разработке приложений: использование микрофронтендов // Инновационное развитие техники и технологий в промышленности: сборник материалов Всероссийской научной конференции молодых исследователей с международным участием, Москва, 16 апреля 2024 года. – М.: Рос. гос. ун-т им. А.Н. Косыгина, 2024. – С. 297–299. – EDN KOYZIT.

2. *Альравашде О.Ю.Б., Горбачев Д.В.* Подход к анализу технологий разработки мобильных приложений // Современные научно-исследовательские и технологические аспекты программной инженерии: материалы Всероссийской научно-технической конференции, Оренбург, 14–15 сентября 2023 года. – Оренбург: Оренбургский государственный университет, 2023. – С. 8–11. – EDN URYTNP.

3. *Сергеев М.Ю., Коробкин А.С.* Подход к разработке информационных систем для интернет- и мобильных приложений // Информационные технологии моделирования и управления. – 2020. – Т. 120, № 3. – С. 234–240. – EDN QXCCNR.

4. *Пивоваров В.В., Хабибуллин Р.М., Нуркаев Р.Р.* BFF – подход к разработке мобильных и веб-приложений, оптимизированный для пользовательского опыта // Вестник Российского нового университета. Серия: Сложные системы: модели, анализ и управление. – 2023. – № 3. – С. 194–201. – DOI: 10.18137/RNU.V9187.23.03.P.194. – EDN ZECPQM.

5. Analysis of cross platform application development over multiple devices using flutter & dart / S. Jadaun, R.K. Singh, R. Kumar, K.K. Agarwal // International Journal of Recent Technology and Engineering (IJRTE). – 2023. – Vol. 12 (1). – p. 33–38. – DOI: 10.35940/ijrte.A7580.0512123.

6. *Чурсин А.Н., Мамедова Н.А., Нефедов Ю.В.* Разработка кроссплатформенных мобильных приложений – перспективные методы и стандартные практики // Прикладная информатика. – 2021. – Т. 16, № 6. – С. 84–102. – DOI: 10.37791/2687-0649-2021-16-6-84-102.

7. Accelerating cross-platform development with flutter framework / A. Md. Sattar, P. Soni, M.K. Ranjan, A. Kumar, C. Sahu, S. Saxena, P. Chaudhari // Journal of Open Source Developments. – 2023. – Vol. 10 (2). – DOI: 10.37591/joosd.v10i2.580.

8. Flutter architectural overview: website. – URL: https://docs.flutter.dev/resources/architectural-overview (accessed: 03.03.2025).

9. *Березовский Н.* Flutter: архитектура фреймворка. Виды сборки. – URL: https://education.yandex.ru/handbook/flutter/article/flutter-arhitektura-frejmvorka.-vidy-sborki (дата обращения: 03.03.2025).

10. *Задябин И.* Flutter: плюсы и минусы использования кросс-платформенной технологии. – 2023. – URL: https://tproger.ru/articles/flutter-plyusy-i-minusy-ispolzovaniya-kross-platformennoj-tehnologii (дата обращения: 03.03.2025).

11. *Колтыгин Д.С., Авсиевич А.В., Седельников И.А.* Аппаратно-программный комплекс для управления робототехническими комплексами // Мехатроника, автоматизация и управление на транспорте: материалы III Всероссийской научно-практической конференции, Самара, 26–27 января 2021 года. – Самара, 2021. – С. 107–111. – EDN KQTDCI.

12. *Колтыгин Д.С., Седельников И.А.* Разработка системы команд для управления роботом-манипулятором // Системы. Методы. Технологии. – 2020. – № 1 (45). – С. 53–60. – DOI: 10.18324/2077-5415-2020-1-53-60. – EDN VAXHSQ.

13. Conditional Importing – How to compile for all platforms in Flutter // Gonçalo Palma Blog. – 2021, October 22. – URL: https://gpalma.pt/blog/conditional-importing/ (accessed: 03.03.2025).

14. Abstract class in dart: learn to code with reusability // BigKnol Blog. – 2024, 30 January. – URL: https://bigknol.com/dart/abstract-class-in-dart-learn-to-code-with-reusability/ (accessed: 03.03.2025).

15. *Колтыгин Д.С., Седельников И.А.* Методика разработки web-приложения для управления робототехническими комплексами // Вестник Астраханского государственного технического университета. Серия: Управление, вычислительная техника и информатика. – 2024. – № 1. – С. 56–63. – DOI: 10.24143/2072-9502-2024-1-56-63. – EDN UPNYHK.

16. *Колтыгин Д.С., Седельников И.А.* Разработка кроссплатформенного приложения для управления робототехническим комплексом // Вестник Астраханского государственного технического университета. Серия: Управление, вычислительная техника и информатика. – 2024. – № 3. – С. 17–25. – DOI: 10.24143/2072-9502-2024-3-17-25. – EDN WALHSN.

*Колтыгин Дмитрий Станиславович*, кандидат технических наук, доцент кафедры управления в технических системах Братского государственного университета. Основное направление научных исследований – автоматика, робототехника. Имеет более 70 печатных работ и учебных пособий. E-mail: kds@brstu.ru

*Седельников Илья Андреевич*, доцент кафедры управления в технических системах Братского государственного университета. Основное направление научных исследований – автоматика, робототехника, программирование. Имеет более 30 печатных работ и учебных пособий. E-mail: Ohtargil@yandex.ru

*Крумин Олег Казимирович*, кандидат технических наук, доцент кафедры управления в технических системах Братского государственного университета. Основное направление научных исследований – автоматика, математический анализ. Имеет более 30 печатных работ и учебных пособий. E-mail: kruminok1977@gmail.com

*Koltygin Dmitry S.*, PhD (Eng.), Associate Professor at the Department of Management in Technical Systems in Bratsk State University. The main area of his scientific research is automation and robotics. He has more than 70 publications and textbooks. E-mail: kds@brstu.ru

*Sedelnikov Ilya A.*, Associate Professor at the Department of Management in Technical Systems in Bratsk State University. The main area of his scientific research is automation, robotics, and programming. He has more than 30 publications and textbooks. E-mail: Ohtargil@yandex.ru

*Krumin Oleg K.*, PhD (Eng.), Associate Professor at the Department of Management in Technical Systems in Bratsk State University. The main area of scientific research is automation and mathematical analysis. He has more than 30 publications and textbooks. E-mail: kruminok1977@gmail.com

DOI: 10.17212/2782-2001-2025-1-7-26

## *Development of a cross-platform application for controlling a robotic complex using Flutter*[*]

*D.S. KOLTYGIN[a], I.A. SEDELNIKOV[b], O.K. KRUMIN[c]*

*Bratsk State University, 40 Makarenko Street, Irkutsk, 665709, Russian Federation*

*[a] kds@brstu.ru [b] Ohtargil@yandex.ru [c] kruminok1977@gmail.com*

**Abstract**

The article provides a brief analysis of approaches to the development of process control programs. The main ones are native applications, web applications, and cross-platform applications. Their main advantages, disadvantages, application and development features are revealed. The description of the Flutter cross-platform technology used in the development of control systems for a robotic complex is presented, including its architecture, the features of the Dart programming language, and the difficulties encountered in implementing program functions related to ensuring software operability on various operating systems. The Flutter framework supports

[*] *Received 10 November 2024.*

programming for Windows, Linux, Mac, iOS, and Android operating systems, as well as for web applications. The application development methodology is presented using the example of a robotic complex with several robots connected to a controller based on an Arduino Mega board and various controls. The structure of the robotics complex and the developed program is described, the appearance of the program on various platforms and a brief description of its operation are given. The issue of user rights management with authentication directly on the controller is considered separately, which allows user authentication on several types of devices. The features of the implementation of cross-platform functions related to the simultaneous use of the dart:io and dart:html libraries in various situations are given using the example of working with a serial port. Algorithms of these processes and justifications of their application are formed. Based on the analysis of this and previous works, the advantages and disadvantages of such development environments as Visual Basic, MIT App Inventor, ASP.Net, MAUI/Xamarin, Flutter, identified during the creation of the software package are presented.

**Keywords**: Cross-platform application, web application, Flutter, Dart, Robot, control system, algorithm, development, SCADA, Arduino

## REFERENCES

1. Isakov L.A. [Innovative approaches in application development: using micro frontends]. *Innovatsionnoe razvitie tekhniki i tekhnologii v promyshlennosti* [Innovative development of technology and engineering in industry]. Collection of materials of the All-Russian scientific conference of young researchers with international participation. Moscow, 2024, pp. 297–299. (In Russian).

2. Al'ravashde O.Yu.B., Gorbachev D.V. [An approach to analyzing mobile application development technologies]. *Sovremennye nauchno-issledovatel'skie i tekhnologicheskie aspekty programmnoi inzhenerii* [Modern research and technological aspects of software engineering]. Proceedings of the All-Russian scientific and technical conference, Orenburg, September 14–15, 2023, pp. 8–11. (In Russian).

3. Sergeev M.Yu., Korobkin A.S. Podkhod k razrabotke informatsionnykh sistem dlya internet- i mobil'nykh prilozhenii [An approach to the development of information systems for Internet and mobile applications]. *Informatsionnye tekhnologii modelirovaniya i upravleniya*, 2020, vol. 120, no. 3, pp. 234–240. (In Russian).

4. Pivovarov V.V., Khabibullin R.M., Nurkaev R.R. BFF – podkhod k razrabotke mobil'nykh i veb-prilozhenii, optimizirovannyi dlya pol'zovatel'skogo opyta [BFF – a user experience-optimized approach to mobile and web development]. *Vestnik Rossiiskogo novogo universiteta. Seriya: Slozhnye sistemy: modeli, analiz i upravlenie = Vestnik of Russian new university. Series: Complex systems: models, analysis, management*, 2023, no. 3, pp. 194–201. DOI: 10.18137/RNU.V9187.23.03.P.194.

5. Jadaun S., Singh R.K., Kumar R., Agarwal K.K. Analysis of cross platform application development over multiple devices using flutter & dart. *International Journal of Recent Technology and Engineering (IJRTE)*, 2023, vol. 12 (1), pp. 33–38. DOI: 10.35940/ijrte.A7580.0512123.

6. Chursin A.N., Mamedova N.A., Nefedov Yu.V. Razrabotka krossplatformennykh mobil'nykh prilozhenii – perspektivnye metody i standartnye praktiki [Development of cross-platform mobile applications – promising methods and standard practices]. *Prikladnaya informatika = Journal of Applied Informatics*, 2021, vol. 16, no. 6. DOI: 10.37791/2687-0649-2021-16-6-84-102.

7. Sattar A. Md., Soni P., Ranjan M.K., Kumar A., Sahu C., Saxena S., Chaudhari P. Accelerating cross-platform development with flutter framework. *Journal of Open Source Developments*, 2023, vol. 10 (2). DOI: 10.37591/joosd.v10i2.580.

8. *Flutter architectural overview*. Website. Available at: https://docs.flutter.dev/resources/architectural-overview (accessed 03.03.2025).

9. Berezovskii N. *Flutter: arkhitektura freimvorka. Vidy sborki* [Flutter: the architecture of the framework. Types of assembly]. Available at: https://education.yandex.ru/handbook/flutter/article/flutter-arhitektura-frejmvorka.-vidy-sborki. (accessed 03.03.2025).

10. Zadyabin I. *Flutter: plyusy i minusy ispol'zovaniya kross-platformennoi tekhnologii* [Flutter: The pros and cons of using cross-platform technology]. 2023. Available at: https://tproger.ru/articles/flutter-plyusy-i-minusy-ispolzovaniya-kross-platformennoj-tehnologii (accessed 03.03.2025).

11. Koltygin D.S., Avsievich A.V., Sedelnikov I.A. [Hardware and software complex for controlling robotic complexes]. *Mekhatronika, avtomatizatsiya i upravlenie na transporte* [Mechatronics,

automation and control in transport]. Proceedings of the III All-Russian scientific and practical conference, Samara, January 26–27, 2021, pp. 107–111. (In Russian).

12. Koltygin D.S., Sedelnikov I.A. Razrabotka sistemy komand dlya upravleniya robotom-manipulyatorom [Development of a command system for controlling a robotic arm]. *Sistemy. Metody. Tekhnologii = Systems Methods. Technologies*, 2020, no. 1 (45), pp. 53–60. DOI: 10.18324/2077-5415-2020-1-53-60. (In Russian).

13. Conditional Importing – How to compile for all platforms in Flutter. *Gonçalo Palma Blog*, 2021, October 22. Available at: https://gpalma.pt/blog/conditional-importing/ (accessed 03.03.2025).

14. Abstract class in dart: learn to code with reusability. *BigKnol Blog*, 2024, 30 January. Available at: https://bigknol.com/dart/abstract-class-in-dart-learn-to-code-with-reusability/ (accessed 03.03.2025).

15. Koltygin D.S., Sedelnikov I.A. Metodika razrabotki web-prilozheniya dlya upravleniya robototekhnicheskimi kompleksami [Web application developing methodology for managing robotic complexes]. *Vestnik Astrakhanskogo gosudarstvennogo tekhnicheskogo universiteta. Seriya: Upravlenie, vychislitel'naya tekhnika i informatika = Vestnik of Astrakhan State Technical University. Series: Management, computer science and informatics*, 2024, no. 1, pp. 56–63. DOI: 10.24143/2072-9502-2024-1-56-63.

16. Koltygin D.S., Sedelnikov I.A. Razrabotka krossplatformennogo prilozheniya dlya upravleniya robototekhnicheskim kompleksom [Development of a cross-platform application for controlling a robotic complex]. *Vestnik Astrakhanskogo gosudarstvennogo tekhnicheskogo universiteta. Seriya: Upravlenie, vychislitel'naya tekhnika i informatika = Vestnik of Astrakhan State Technical University. Series: Management, computer science and informatics*, 2024, no. 3, pp. 17–25. DOI: 10.24143/2072-9502-2024-3-17-25.

Для цитирования:

*Колтыгин Д.С., Седельников И.А., Крумин О.К.* Разработка кросс-платформенного приложения для управления робототехническим комплексом с помощью Flutter // Системы анализа и обработки данных. – 2025. – № 1 (97). – С. 7–26. – DOI: 10.17212/2782-2001-2025-1-7-26.

For citation:

Koltygin D.S., Sedelnikov I.A., Krumin O.K. Razrabotka kross-platformennogo prilozheniya dlya upravleniya robototekhnicheskim kompleksom s pomoshch'yu Flutter [Development of a cross-platform application for controlling a robotic complex using Flutter]. *Sistemy analiza i obrabotki dannykh = Analysis and Data Processing Systems*, 2025, no. 1 (97), pp. 7–26. DOI: 10.17212/2782-2001-2025-1-7-26.